

工业机器人运动碰撞的仿真实现

陈胜奋, 谢明红

(华侨大学 机电及自动化学院, 福建 厦门 361021)

摘要: 在对工业机器人模型进行碰撞检测时,提出一种把 AABB 和 OBB 结合的改进包围盒法.首先,在各个杆件模型的坐标系下,利用八叉树结构建立杆件模型的改进层次包围盒;其次,把在包围盒内的模型三角面序号存放在对应叶子节点的数组里;将包围盒位姿转换到基坐标系下的位姿,利用 OBB 包围盒的分离轴理论进行碰撞检测;接着,利用叶子节点里的索引序号索引到对应三角面并进行精确的三角面求交;最后,生成一个应用程序并进行仿真.实验结果表明:生成的应用程序能够快速和精确地检测出模型碰撞,且运行流畅.

关键词: 包围盒; 分离轴; 碰撞检测; 叶子节点

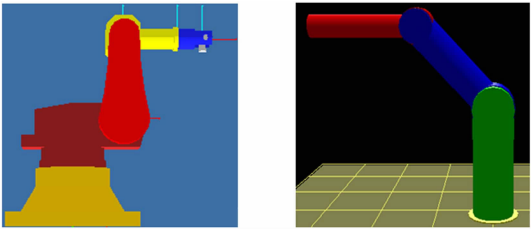
中图分类号: TG 659 **文献标志码:** A

工业机器人工作的范围广、灵活性强等特点使其在自动化领域得到了广泛的应用^[1].但工业机器人在运动过程中容易发生碰撞,碰撞的后果十分严重^[2].因此,工业机器人的碰撞检测是十分必要的.目前,对碰撞检测普遍的做法是以包围盒碰撞检测进行快速粗检测,然后以三角形面片求交进行精检测.包围盒法的思想是用略大且规则的简单包围盒代替复杂的模型.包围盒(长方体)模型主要有 AABB、OBB 等,每一种包围盒都具有各自的优势和不足^[3].对包围盒进行粗略的碰撞检测可以节省时间^[4-5].当前,对包围盒碰撞的研究有很多. Gottschalk 等^[6]采用 OBB 树进行碰撞检测研究; Wang 等^[7]采用 AABB 包围盒近似表述模型并粗略判断碰撞的位置; Chang 等^[8]结合包围盒和包围球进行快速碰撞检测.但是,很少有对包围盒本身进行研究.在综合考虑层次包围盒检测效率下,本文提出一种改进的层次包围盒法的碰撞检测方法,以提高检测速度.

1 工业机器人模型的建立

工业机器人的碰撞检测是判断各个杆件之间是否存在碰撞或干涉,其中首要任务是建立工业机器人模型.建立模型的方法一般有:1)使用 glut 库函数代码编程,由于 glut 库函数只有一些简单的三维模型,如圆柱,长方体等,这样生成的模型也就相对简单;2)使用三维建模软件,如 UG,PRO/E,这些软件可以建立各种各样的模型且操作简便.相比采用第二种方法生成的模型可以达到更好的仿真效果.

由三维建模软件建立各个杆件模型并将模型保存为 3ds 格式文件,再由 VC++ 代码编程,读取杆件模型的文件,通过 VC++ 代码将各个杆件进行一系列的平移和旋转,变换到空间相应位置并用 OpenGL 显示^[9-10],以实现工业机器人的模型建立.工业机器人两种不同的建模方式,如图 1 所示.比较图 1(a),1(b)可知:三维建模软件建立的工业机



(a) 结合 PRO/E 和 OpenGL 建模 (b) 使用 glut 库建模

图 1 工业机器人两种不同的建模方式

Fig. 1 Two different modeling methods of building industrial robot

器人模型可以相对复杂且更有效率.

2 改进层次包围盒的建立

2.1 改进包围盒的建立

构造 AABB 包围盒计算量较少,只需比较模型对象各点元素的坐标值,就可以得到坐标轴上的最大值与最小值.这些值构成了 AABB 包围盒的顶点(图 2(a)).OBB 包围盒计算则相对复杂,首先得通过这些点元素的坐标值计算出协方差矩阵;然后计算协方差矩阵的特征值和特征向量;最后以特征向量为新的坐标轴,把所有点元素的坐标投影到新的坐标轴上,得到在新坐标轴上坐标最大值和最小值,这些值构成了 OBB 包围盒的顶点(图 2(b)).

综上可知,AABB 包围盒计算简单,但结构不紧凑;OBB 包围盒结构紧凑,但计算复杂.因此结合两者的优点,提出了一种基于 AABB-OBB 结合改进的包围盒(图 2(c)).程序仿真的工业机器人模型是由代码读取每个独立的杆件模型文件并显示,以实现快速建模.每个杆件模型都有各自的坐标系,每个包围盒都是基于对应杆件模型的坐标系建立的.选择合适的杆件坐标系是十分必要的,它可以减少包围盒建立的计算量,一般选择杆件旋转中心为坐标系原点.各个杆件模型的坐标系是独立的,不相互影响,所以构造的包围盒只需在对应杆件模型的坐标系下,通过比较点元素坐标最值,由最值构造出改进的包围盒顶点.改进的包围盒除了坐标系与原来包围盒坐标系不同外,其他的步骤与构造传统 AABB 包围盒相同.由图 2 可知:改进的包围盒既有 OBB 包围盒的紧凑,又有 AABB 包围盒构造的简便.

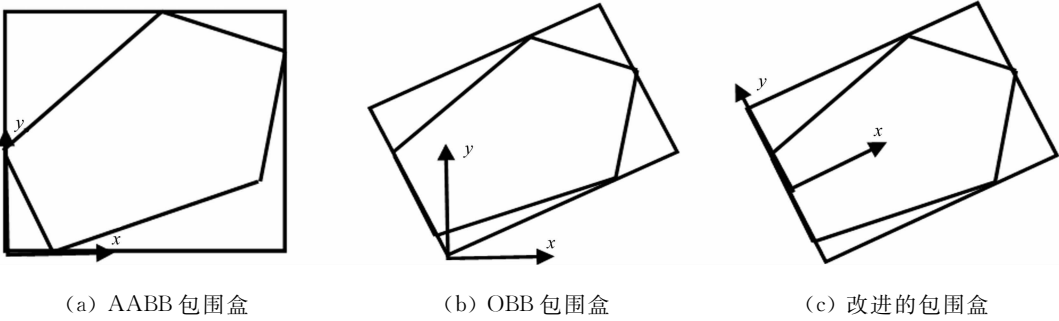


图 2 包围盒的二维示意图
Fig. 2 2D schematic diagram of bounding box

工业机器人作业时,杆件模型会发生旋转,各模型点元素坐标就会变化,构造的包围盒也会随着变化.AABB 或 OBB 采用重构的方法是先重新计算新的杆件模型点元素坐标,再由原先构造包围盒的步骤进行重构;而改进的包围盒针对这种情况采用的方法比较灵活,只需计算杆件模型坐标系间的变换矩阵,无需重新计算点元素坐标.较传统包围盒重构,改进的包围盒重构减少了大量运算,提高了效率.

2.2 八叉树结构的建立

八叉树是层次包围盒进行有效快速分割的一种方法^[11].从根节点出发将根节点分割成 8 个节点,再将节点分割成新的 8 个节点.重复此步骤直至满足分割深度条件时,把包围盒内的三角形序号存储到对应的叶子节点数组内.八叉树结构的示意图和建立流程图,分别如图 3,4 所示.



图 3 八叉树结构示意图
Fig. 3 Octree structure diagram

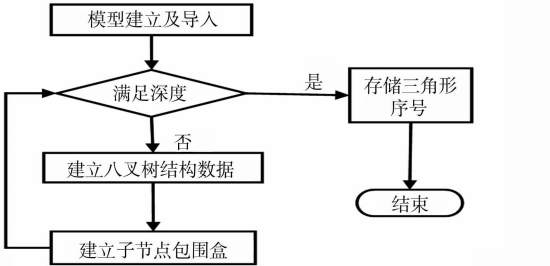


图 4 八叉树建立的流程图
Fig. 4 Flow chart of establishing octree

构建层次包围盒模块需要构造一个合适的存储数据结构,以便代码程序调用.一般构造层次包围

盒采用的数据结构有链表和数组. 定义数组时, 数组长度大小必须是已知的, 但包围盒的层数是变化的, 很容易造成数组的越界或者内存空间的浪费. 链表采用指针的方式, 可以动态申请内存空间, 需要时就向系统申请内存, 不需要时也可以动态删除. 这样既避免了越界的风险, 又提高了对内存空间的利用.

定义存储数据结构如下

```
Class COctree{public:
    int maxdepth;//包围盒的层数,最小值为 0
    vector<int>index;//对应包围盒内的三角形(面)的序号索引
    float xmin,xmax,//x 坐标的最小值和最大值
    ymin,ymax,//y 坐标的最小值和最大值
    zmin,zmax;//z 坐标的最小值和最大值
    void setSize(COctree * root,int maxdepth,float xmin,float xmax,
    float ymin,float ymax,float zmin,float zmax);//设置包围盒的大小
    COctree * chldnodes[8];//八个节点的指针
}
```

3 碰撞检测

3.1 粗略检测

依据分离轴理论^[12], 不同杆件模型的包围盒碰撞检测需在同一个坐标系下进行. 由于改进的包围盒都是基于对应杆件模型的坐标系, 所以不同杆件模型坐标系之间的相对位姿就显得格外重要. 工业机器人模型初始的位置和姿态信息保存在装配文件里, 通过 VC++ 编程读取装配文件获得所需的位置和姿态, 通常位姿信息是用 4 维方阵表示. 由杆件模型的初始位姿信息和工业机器人运动学性质^[13], 可以得出 2 个不同杆件模型 a, b 之间的相对位置和姿态关系, 即

$$\mathbf{M}_a^b = \mathbf{M}_a^{a+1} \times \mathbf{M}_{a+2}^{a+3} \times \cdots \times \mathbf{M}_{b-1}^b = \begin{bmatrix} \mathbf{T}_a^b & \mathbf{l}_a^b \\ 0 & 1 \end{bmatrix}. \quad (1)$$

式(1)中: \mathbf{M}_a^b 为杆件模型 a 和 b 之间的相对位置和姿态关系; \mathbf{T}_a^b 为 3×3 姿态矩阵; \mathbf{l}_a^b 为位置向量.

在进行检测前, 需将不同包围盒的中心位置转换到同一个坐标系下. 包围盒间的中心位置随着杆件模型运动变化, 而包围盒中心坐标相对于对应的模型坐标系是不变的. 因此, 只需知道模型坐标系间的相对位姿关系, 就可以快速将改进包围盒中心位置转换到同一坐标系下. 式(1)的相对位姿是两杆件模型的旋转点之间的位姿. 杆件模型旋转中心为杆件坐标系原点, 在杆件模型坐标系下, 包围盒的相对姿态 \mathbf{T}_a^b 不随点移动而变化. 可以得到不同包围盒的中心相对姿态, 即

$$\mathbf{M}_{ac}^{bc} = \begin{bmatrix} \mathbf{T}_a^b & -\mathbf{i} + \mathbf{j} \\ 0 & 1 \end{bmatrix}. \quad (2)$$

式(2)中: \mathbf{M}_{ac}^{bc} 包围盒中心点相对位姿; \mathbf{i} 为杆件模型 a 的中心在自身坐标系下的位置矢量; \mathbf{j} 为杆件模型 b 的中心在 a 坐标系下的位置矢量.

把不同包围盒坐标转换到同一坐标系后, 依据分离轴理论, 经过 15 次投影判断, 就可以快速判断包围盒是否碰撞.

3.2 精确检测

八叉树层数越多, 包围盒就越能准确地包围模型, 碰撞检测结果也就越近似模型碰撞结果. 但如果仅依靠增加八叉树层数来提高判断精度, 模型碰撞会产生误判, 使正常加工难以进行. 这是因为采用层次包围盒并不能精确表示模型, 包围盒碰撞检测只能确定可能发生碰撞的区域, 并不能确定工业机器人模型在该区域发生碰撞. 当八叉树层数超过一定数目时, 系统会占用很大的内存, 造成程序不能顺畅运行. 所以, 判断模型是否产生碰撞, 不能仅通过增加八叉树层数, 而需要新的方法来进行精确碰撞检测.

虽然机器人造型的软件各不相同, 但一般通过软件建立的模型都是由三角网格面构成的. 模型的三角网格就可以精确表示模型, 所以对模型进行精确的碰撞检测就可以变成对不同杆件模型之间的三角

网格求交的问题. 只要三角面片相交, 对应的两个模型就会发生碰撞.

粗略碰撞检测确定了碰撞区域后, 可以通过索引直接获得区域的三角形序号 i 及三角形的三个顶点坐标 p_0, p_1, p_2 ; 然后采用 Devillers 算法, 即通过两个三角形的顶点构成的行列式的正负来判断点与面之间的相对位置; 接着计算与平面的交点的位置是否在三角形内, 判断两个三角形是否产生干涉, 流程如图 5 所示.

求线段两端点相对三角形的位置函数为 `bool vector4_det(float3 p0, float3 p1, float3 p2, float3 p3, float3 p4)`; 求线段与三角形交点的位置函数为 `bool ValidPoint(float3 p0, float3 p1, float3 p2, float3 p3, float3 p4)`. 其中, p_0, p_1, p_2 是 $\triangle p_0 p_1 p_2$ 的三个顶点坐标; p_3, p_4 是 $\triangle p_3 p_4 p_5$ 两顶点 p_3 和 p_4 坐标. 通过函数 `vector4_det` 的返回值来判断线段 $p_3 p_4$ 的两端点相对 $\triangle p_0 p_1 p_2$ 的位置, 如果是 true 就表示 p_3 和 p_4 在 $\triangle p_0 p_1 p_2$ 异侧; 否则, 就是同侧. 通过函数 `ValidPoint` 的返回值判断线段与三角形交点的位置, 如果是 false, 则线段 $p_3 p_4$ 与 $\triangle p_0 p_1 p_2$ 的交点不在 $\triangle p_0 p_1 p_2$ 内, $\triangle p_3 p_4 p_5$ 中的线段 $p_4 p_5$ 和 $p_3 p_5$ 重复判断线段 $p_3 p_4$ 的过程; 如果是 true, 则 $\triangle p_0 p_1 p_2$ 和 $\triangle p_3 p_4 p_5$ 相交, 可判定杆件模型产生碰撞. 重复区域内所有的三角形之间的干涉检测, 只要有一次判断干涉, 工业机器人就产生碰撞.

4 软件仿真实实现实例

定义 `ModelContext` 类存储读入的杆件的基本属性; 定义 `GSMatrix4X4_f` 四维矩阵存储杆件的位置和姿态; 定义 `COctree` 类存储八叉树的子节点、深度、索引号等信息; 定义 `CVector` 类存储点坐标和矢量坐标等.

应用 `VC++` 和 `OpenGL` 进行软件编程. 读入杆件模型文件和装配信息文件, 把基本信息属性存储在 `ModelContext` 类, 把杆件模型的位置和姿态存储在 `GSMatrix4X4_f` 类. 根据 `ModelContext` 类和 `GSMatrix4X4_f` 类的数据信息建立改进的层次包围盒(图 6(a)), 并把包围盒的数据存储在 `COctree` 类里. 依据 `COctree` 类和 `GSMatrix4X4_f` 类的数据信息可以进行粗略的碰撞检测(图 6(b)). 精确碰撞检测(图 6(c))则需要 `ModelContext` 类、`COctree` 类和 `GSMatrix4X4_f` 类的数据, 得出碰撞的精确检测. 通过时间测量, 构造 AABB 包围盒并完成精确检测约需要 5.6 ms, 构造 OBB 包围盒并完成精确检测约需要 5.5 ms, 而构造改进的包围盒并完成精确检测约需要 5 ms, 减少了检测时间.



图 6 软件实现的过程示意图

Fig. 6 Schematic diagram of the software implement

5 结束语

采用三维建模软件建立工业机器人模型, 用 `VC++` 编程读取模型文件并显示, 这种方法可以快速建立比较复杂的模型. 首先, 在各自模型的坐标系下建立对应的包围盒; 其次, 通过坐标系间的变换关系, 把包围盒坐标转换到同一个坐标系下, 再应用分离轴理论进行包围盒碰撞检测; 然后, 利用三角形求交

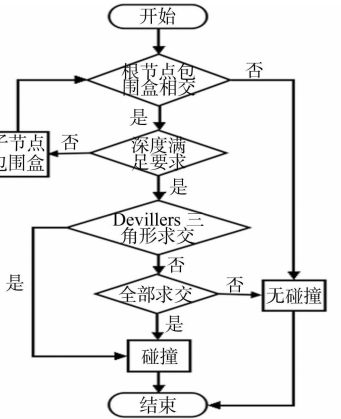


图 5 碰撞检测的流程图
Fig. 5 Flow charts of collision detection

的方法进行精确检测;最后,生成一个实例程序完成碰撞检测的整个过程.通过时间测量,改进的算法提高了检测速度.

参考文献:

[1] 孙志杰,王善军,张雪鑫.工业机器人发展现状与趋势[J].吉林工程技术师范学院学报,2011,27(7):61-62.

[2] 刘雄伟.基于三角面片检测的五轴数控加工碰撞干涉检测智能算法研究[D].广州:广东工业大学,2011:1-4.

[3] 邹益胜,丁国富,许明恒,等.实时碰撞检测算法综述[J].计算机应用研究,2008,25(1):8-10.

[4] GARCIA-ALONSO A,SERRANO N,FLAUER J. Solving the collision detection problem [J]. IEEE Computer Graphics and Applications,1994,14(3):36-43.

[5] GOTTSCHALK S. Collision queries using oriented bounding boxes[D]. North Carolina:University of North Carolina at Chapel Hill,2000:64-86.

[6] GOTTSCHALK S,LIN M,MANOCHA D. OBBtree: A hierarchical structure for rapid interference detection[C]// Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. New Orlean:ACM,1996:171-180.

[7] WANG Yao,HU Yan-juan,FAN Jiu-chen. Collision detection based on bounding box for NC machining simulation [J]. Physics Procedia,2012,24(1):247-252.

[8] CHANG J W,KIM M S. Efficient triangle-triangle intersection test for OBB-based collision detection[J]. Computer and Graphics,2009,33(3):235-240.

[9] 侯俊杰.深入浅出 MFC[M]. 2 版. 武汉:华中科技大学出版社,2001:309-334.

[10] 牛书涛,库祥臣.基于 Visual C++ 与 OpenGL 的数控车削加工仿真系统研究[J].机电一体化,2012,8(3):35-39.

[11] 吴明华,余勇翔,周济.采用空间分割技术的八叉树干涉检验算法[J].计算机学报,1997,20(9):849-854.

[12] 吴峰,于东,张晓辉,等.数控机床实时碰撞检测算法的研究与实现[J].组合机床与自动化加工技术,2011(12):49-51.

[13] 于靖军,刘辛军,丁希仑,等.机器人机构学的数学基础[M].北京:机械工业出版社,2008:278-281.

Simulation of Industrial Robot Motion Collision

CHEN Sheng-fen, XIE Ming-hong

(College of Mechanical Engineering and Automation, Huaqiao University, Xiamen 361021, China)

Abstract: In the collision detection of the industrial robot model, an improved octree method based on combination of AABB and OBB is put forward. Firstly, in various rod model coordinates, each improved bounding box of manipulator model by octree is built; secondly, the triangles numbers in the model box are stored in an array of corresponding leaf nodes; and the pose of bounding box is translated to the pose in the base coordinates systems, then collision detection is done using OBB separation axis theory; next, the corresponding triangle in the leaf nodes is found by index ordinal, then triangular surface precise intersection is done; finally, an application is built and simulated. The results have proven that the application can quickly and accurately detect the collision and run smoothly.

Keywords: bounding box; separation axis; collision detection; leaf node

(责任编辑:黄晓楠 英文审校:杨建红)